Homework Assignment #8
Due: November 14, 2025 at 11:59 p.m.

1. We looked at Dijkstra's solution (restated below) to the N-process mutual exclusion problem in class. We assume the processes are numbered $1..N$.

```
1:  Shared variables:
2:  Boolean Interested[1..N]        ▷ which processes are interested in the shared resource
3:  Boolean Passed[1..N]            ▷ which processes have passed Phase 1
4:  integer K                       ▷ the label of a process allowed to try for the resource next
5:  Local variable:
6:  Boolean done                    ▷ determines when the process can exit the outer repeat loop
7:  Code for process i:
8:  Interested[i] ← true
9:  repeat
10:     while K ≠ i do              ▷ Entry Phase 1
11:         if not Interested[K] then ▷ try to go next
12:             K ← i
13:     Passed[i] ← true            ▷ begin Entry Phase 2
14:     done ← true
15:     for j ← 1..N except i do    ▷ check if anyone else has passed Phase 1
16:         if Passed[j] then       ▷ go back to Phase 1
17:             Passed[i] ← false
18:             done ← false
19: until done
20: CRITICAL SECTION
21: Passed[i] ← false
22: Interested[i] ← false
23: REMAINDER SECTION
```

For the critical section, we assume that any process that enters the critical section (and does not experience a stopping failure) eventually completes the critical section and begins executing its exit section.

The original paper lists many properties that the algorithm is supposed to satisfy, but does not really explain why all of them hold. For example, property (c) says "If any of the computers is stopped well outside its critical section, this is not allowed to lead to potential blocking of the others" and this point is not explicitly addressed in the proof of correctness. Let's examine what "well outside" means for this algorithm. The system is said to be *blocked* in an infinite execution if there is some suffix of the execution where some process performs infinitely many steps of the entry section and no process ever enters the critical section. Give a precise description of the conditions when a stopping failure of a process could cause the system to block. (For example, a possible answer might be that blocking could occur if and only if a process stops when it is just about to execute line 8 or line 14, and there is a full moon.)

Briefly explain why your answer is correct. That is, explain how stopping under the conditions you give could cause the system to block, and explain why stopping in any circumstance that does not satisfy your conditions cannot cause the system to block.